International Journal of Innovative Computing, Information and Control
Volume **13**, Number **2**, April **2017**

ICIC International ©2017 ISSN 1349-4198
pp. 397–409

IMPLEMENTATION OF EYE DETECTION USING DUAL CAMERA ON THE EMBEDDED SYSTEM

Aryuanto Soetedjo and I Komang Somawirata

Department of Electrical Engineering
National Institute of Technology
Jalan Raya Karanglo KM 2 Malang, Indonesia
aryuanto@gmail.com; kmgsomawirata@yahoo.com

Received September 2016; revised January 2017

Abstract. *This paper presents an implementation of eye detection on the embedded system. Two camera systems based on the low cost Raspberry Pi modules are employed. To speed up the process, the proposed system implements the face detection technique and the eye detection technique on two camera modules separately. The face detection module detects the bounding box of face and sends the coordinates to the eye detection module via a serial communication. In the eye detection module, the eye is searched on a limited area defined by the face's bounding box. The popular Viola-Jones object detection is employed in the face detection module. Three eye detection techniques, consisting of the Viola-Jones method, the eye-map method, and the Hough circle transform method, are implemented and evaluated in the eye detection module. The best result is obtained by the Hough circle transform method, where the frame rate of 30.020 fps, the true positive rate of 0.869, and the precision of 0.824 are achieved.*

Keywords: Face detection, Eye detection, Viola-Jones, Eye-map, Hough transform, Raspberry Pi, Dual camera

1. **Introduction.** The camera-based systems play significant roles in the recent trend in technology. Many applications adopt the vision systems for replacing the conventional techniques, or developing the new techniques. One of them is an application for detecting the driver fatigue as proposed in [1-15]. A typical method developed in [1-10] is by analyzing the eye state (eye is open or close). The combination of eye state and other features is also commonly used, such as the head gesture [11], and the yawning [12-15].

Several methods are employed to detect the eye such as the Haar classifiers method, which is sometimes called as the Viola-Jones method (VJ) [1-4,11,12,16,17], using infrared camera [5-8], the Hough transform for detecting the circle [9], the color segmentation [14,19-21], and the projection technique [22].

The VJ is a popular technique for detecting the objects. The technique relies on the integral image, the Adaboost learning algorithm, and the cascade mechanism [26]. In [2], the VJ was employed to extract the eye region from the face image. Then the shape of eye was calculated to determine the open or close eye. The optimal sampling ratio and the minimal object size of the eye detection using the VJ was discussed in [3]. To limit the search area, the VJ was applied on a defined region based on the face area [4].

The VJ sometimes fails to detect the eye due to the similar characteristic of other objects on the face, such as eyebrows, nostrils, and mouth. To overcome this problem, the geometric features of the eye were calculated after the detection process [12]. The other method was proposed in [16] by introducing the new rectangle features.

Another approach to detect the eye is using an infrared camera. The basic principle of the technique is that the eye (pupil) reflects the infrared light to produce the bright pupil

effect. The drawback of this method is that the infrared illuminator should be placed along the center of camera axis [5]. The practical implementation is by placing a few infrared illuminators on the concentric ring surrounding the camera. The method proposed in [6] utilized the infrared illuminator in different ways. The infrared illuminator was used to make the brighter face image compared to the background. The eye was detected by comparing the difference image obtaining by subtracting two succeeding frames.

Since the iris of eye forms the circle, the shape detection method using the Hough transform was employed to detect the eye [9]. The Hough transform was applied on the edge image of the search region defined on the face. The eye state is determined by measuring the intersection of the edge image and the obtained circle image.

The eye-map calculated from the chrominance and luminance components of YCbCr color space was constructed to detect the eye [8,14,19,20]. The open eye was determined when the area of binary image obtained from the eye-map is larger than a threshold. The white color segmentation based on the normalized RGB chromaticity diagram was employed to detect the white sclera of eye [21]. Then the ellipse fitting method was applied to obtain the precise boundary of eye.

The position of eye could be found by projecting the edge image of the face [22]. The complexity feature, which is calculated based on the numbers of edges and areas, was used to find the accurate face location.

To be implemented in the real situation, the algorithm of eye detection system should be fast enough. Moreover, for the driver fatigue detection system, it should be implemented on the embedded platform for easy installation on the car. Almost all the works described previously implemented the algorithms using the USB camera or Webcam and personal computer or laptop. Only a few of them implemented it on the embedded platform such as based on the Tiny4412 [3]. The embedded platform discussed here refers to the single board computer system with a capability to perform the image processing tasks. Recently, the Raspberry Pi single board computer systems are commonly adopted, such as used for recognizing the road sign [23], detecting and recognizing the moving objects in the surveillance system [24], and recognizing the person [25].

This paper deals with the real-time implementation of eye detection on the embedded system. The main contribution of our work is in increasing the computational speed by introducing the dual camera system. The proposed dual camera system differs from the existing techniques [14,15,17] in the camera configuration and the hardware platform. The previous methods used two cameras to detect the driver fatigue and the lane [14], detect the face and the mouth [15]. The method in [17] used one wide view camera and one narrow view camera to detect the face and the eye. In the existing methods, two cameras are interfaced to the personal computer for performing the image processing tasks. Therefore, it is not suitable for the real implementation on the car. Our method employs two Raspberry Pi single board computers, where each module is equipped with the camera module. A simple serial communication is adopted to communicate between the modules. The first module is used to detect the face, while the second one is used to detect the eye on the limited search area defined by the first module. Since the detection methods are executed on different processors simultaneously, the computational speed for detecting the eye could be increased significantly. This approach takes advantages of the low cost and the small size of the Raspberry Pi module for an easy implementation on the car.

The rest of paper is organized as follows. Section 2 describes the proposed system. It covers the method and implementation of the system. Section 3 presents the experimental results. Conclusion is covered in Section 4.

2. Proposed System.

2.1. **System overview.** In the eye detection process, it is common to limit the eye search area to increase the eye detection speed and accuracy. The search area is defined based on the detected face area. Thus, the face detection process is carried out first before the eye detection. Unfortunately, it contributes most of the computational burden.

To overcome the problem, we propose a new technique using the dual camera system. The contributions of our research are as follows. First, it performs the face detection and the eye detection on the separated modules. By processing the eye detection separately, the computational speed could be increased. Furthermore, since the search area in the eye detection process is limited only on the face area detected by the face detection module, the computational speed of the eye detection could be improved significantly. Second, the proposed system is implemented on two Raspberry Pi single board computer systems. It provides the low cost and small size embedded systems for easy installation on the vehicle's dashboard to detect the driver's eye. Third, several eye detection algorithms such as the VJ method, the eye-map method, and the Hough circle transform method are implemented to investigate their performances in the detection rates and the frame rates.

The system architecture is illustrated in Figure 1, where the face detection runs on the face detection module, while the eye detection runs on the eye detection module. The system consists of five components: a face detection processing unit, an eye detection processing unit, a face detection camera unit, an eye detection camera unit, and a camera frame. Two camera units are attached side by side on the camera frame.

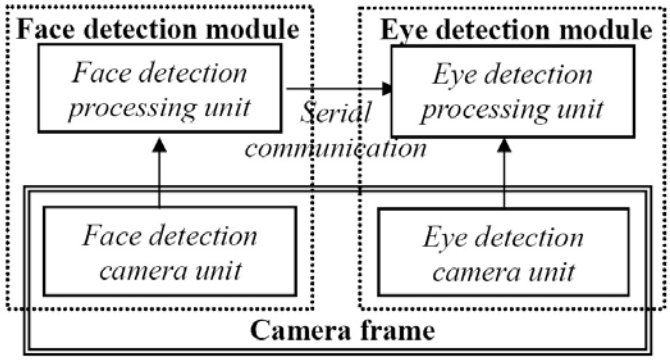

FIGURE 1. Configuration of dual camera system

A serial communication is employed to communicate between the face detection processing unit and the eye detection processing unit. It is a simple communication for sending the coordinates of detected face from the face detection module to the eye detection module. Since two camera units are arranged in a fixed position on the camera frame, the coordinate transformation between both cameras could be calculated easily.

The proposed system works as follows. In the initialization stage, the eye detection module waits for the data sent from the face detection module. During this stage, the eye detection module captures the image, but does not process it. The face detection module captures the image and performs the face detection continuously. When the face is detected, the face detection module sends the coordinates of detected face to the eye detection module. Once receiving the coordinates, the eye detection module starts to perform the eye detection.

The face detection module employs the VJ method to detect the face. It detects the face and defines the coordinates of bounding box of the detected face. Since the face detection camera unit and the eye detection camera unit are located side by side, the coordinates

of detected face should be transformed into the eye camera frame as described in the next section. Then the transformed coordinates are sent to the eye detection module. The coordinates are sent only when the face is detected. When the face is not detected, there are no data sent to the eye detection module.

Upon receiving the coordinates, the eye detection module crops the captured image according to the coordinates and detects the eye. The search area is limited on the face area only. It will speed up the detection process. Once the eye detection starts, it is carried out continuously, even if there are no coordinates updated from the face detection module. In this case, the eye detection module uses the last coordinates.

In this work, we implement three eye detection methods, i.e., the VJ method [1-4,11,12,16,17], the eye-map method [8,14,19,20], and the Hough circle transform method [9] on the eye detection module. Their performances are compared to examine the effectiveness of the proposed approach.

2.2. **Detection method.**

2.2.1. *Transformation of camera's coordinate.* As described previously, the eye detection process is carried out on the eye detection module. The module gets the coordinates of detected face from the face detection module. As shown in Figure 1, since both camera units are installed on a planar frame side by side, the face coordinates detected by the face detection module are different from the one detected by the eye detection module. To transform the face coordinates on the face detection module into the eye detection module, we adopt a simple practical method as follows.

It is noted here that the proposed system is intended to detect the driver's eye for detecting the driver fatigue. It is also assumed that the modules could be installed on a proper position in such way that only one face (the driver) appears on the camera. Figure 2 depicts the model of dual camera system, where I_{fm} and I_{em} are the image planes of face detection and eye detection cameras respectively, W_{obj} is the line denoting the width of face object, O_{fm} and O_{em} are the optical center of face detection and eye detection cameras respectively, TL and TR are the points on left side and right side of face object respectively, tl_{fm} and tl_{em} are the image points of TL on face detection and eye detection

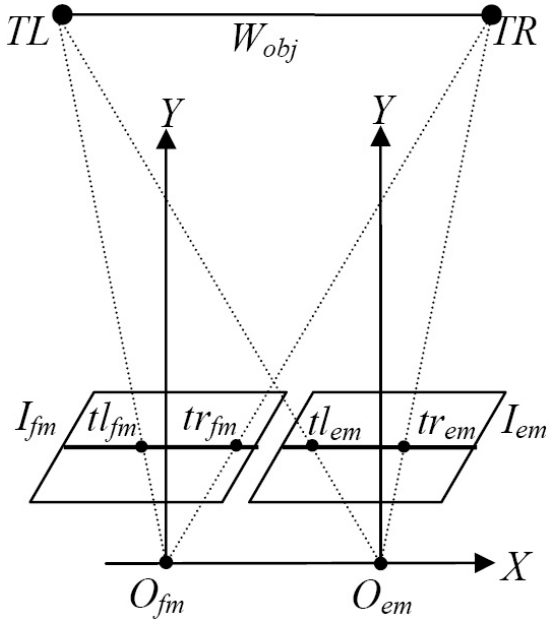

FIGURE 2. Dual camera model

cameras respectively, and tr_{fm} and tr_{em} are the image points of TR on face detection and eye detection cameras respectively.

The installation of camera modules should ensure that the image points of face object lie inside the image planes I_{fm} and I_{em}. From the model shown in Figure 2 and the cameras arrangement in Figure 1, it is clear that the coordinate transformation between the image points on the face detection and eye detection cameras involves the displacement in x-coordinate only. The displacement value is found experimentally in the calibration process that is done once during the hardware installation.

The calibration process is done by implementing the face detection algorithm on both face detection and eye detection modules. Then run them simultaneously to detect the face. Since both modules use the same hardware, the detected faces will be the same, in the sense that the bounding box of detected face obtained by the face detection module occupies the same area with the one obtained by the eye detection module. However, the coordinates of the bounding box on each module are different. By subtracting the x-coordinate of the bounding box (could be one point on the corner of bounding box) on the face detection module by the one on the eye detection module, the displacement value could be obtained.

2.2.2. *Eye detection.* The face detection module is employed to provide the bounding box of face to the eye detection module. The bounding box is defined by the coordinates of four corners. The coordinates are sent out when the face is detected. This bounding box is used by the eye detection module to determine the eye search area.

The proposed eye detection method is depicted in Figure 3. The algorithm starts when it receives the coordinates of face's bounding box from the face detection module. Once the bounding box is received, it is used to define the eye search area until the new update is received. The eye detection module could execute the eye detection algorithm freely, without having to wait for the data (coordinates of face's bounding box) sent by the face detection module. Therefore, the computational time of eye detection module could be decreased, and it is not affected by the computational time of face detection module. This approach works well when the human face does not move rapidly.

2.2.3. *Image resizing.* The relationship among image rescaling, frame rate and detection rate is discussed in [3]. Rescaling image to a smaller one increases the frame rate. However, it reduces the detection rate accordingly. In addition, the optimal scale factor should consider the minimum object size allowed by the VJ, which is usually set to 20×20 pixels, or 30×30 pixels.

In the eye detection module, the eye detection method is applied on the search area which is defined by the face's bounding box. The image resizing discussed in the following refers to this search area, not the whole captured image. We propose two methods for resizing the search area. In the first method, the search area is defined according to (1)-(4), where *sa_top*, *sa_left*, *sa_width*, and *sa_height* are the top border, the left border, the width, and the height of search area respectively; *fb_top, fb_left, fb_width*, and *fb_height* are the top border, the left border, the width, and the height of face's bounding box respectively. This method considers that the eye is located on two thirds upper part of the face as expressed by (4).

$$sa\_top = fb\_top \tag{1}$$

$$sa\_left = fb\_left \tag{2}$$

$$sa\_width = fb\_width \tag{3}$$

$$sa\_height = fb\_height \times 0.67 \tag{4}$$

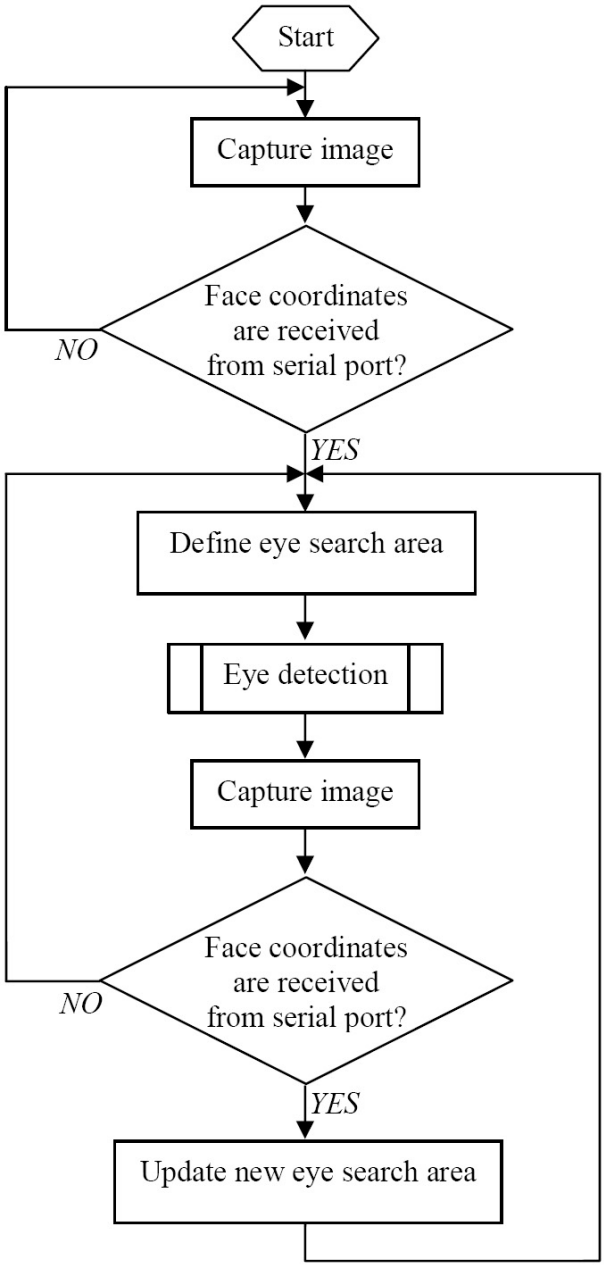

FIGURE 3. Eye detection method

In the second method, the search area is defined in the similar way, but it is rescaled down so that the new search area has a fixed height. It is expressed by (5)-(8), where the height of search area (sa_height) is defined in advance; β is the scaling factor, which is calculated during detection process. In this method, sa_height is defined by considering the minimum eye size allowed by the eye detection method.

$$\beta = sa\_height/(fb\_height \times 0.67) \tag{5}$$

$$sa\_width = fb\_width \times \beta \tag{6}$$

$$sa\_top = fb\_top \times \beta \tag{7}$$

$$sa\_left = fb\_left \times \beta \tag{8}$$

2.3. **Implementation.**

2.3.1. *Hardware implementation.* The proposed system is implemented on the low cost embedded camera systems, namely the Raspberry Pi [27] single board computer systems. Both the face detection processing unit and the eye detection processing unit are implemented using the Raspberry Pi Type 3 Model B modules. The module has a 1.2 GHz 64-bit quad-core ARMv8 CPU, 1GB RAM, 40 GPIO pins, Camera Serial Interface (CSI), microSD card storage. The price of each module is about US$ 35. The board dimension is 85.6mm × 56mm × 21mm.

Both the face detection camera unit and the eye detection camera unit are implemented using the Raspberry Pi camera. The camera has the OmniVision OV5647 image sensor, the sensor resolution of 2592 × 1944 pixels, the horizontal field of view of 53.50^0, the vertical field of view of 41.41^0. The camera dimension is 25mm × 24mm × 9mm. The price of each camera module is about US$ 25. The camera is connected to the Raspberry Pi module via the CSI connector. From a few experiments, and referring to the previous work proposed in [10], the resolution of 320 × 240 pixels is chosen as the optimal resolution used in this experiment. The prototype of hardware system is depicted in Figure 4.

FIGURE 4. Prototype of hardware system

The serial communication between the face detection module and the eye detection module is performed via the UART (universal asynchronous receiver/transmitter) on the GPIO (general purpose input/output) pins. The communication parameters are:

- Data bits: 8 bits,
- Baud rate: 9600 bps,
- Parity: none,
- Stop bits: 1 bit,
- Flow control: none.

2.3.2. *Software implementation.* The Raspberry Pi used in the research runs on the Raspbian operating system [28]. It is a Debian-based operating system for the Raspberry Pi. The proposed algorithm is implemented using C++. The OpenCV library [29] is adopted to handle the image processing tasks.

The OpenCV provides useful libraries used in the experiments, such as capturing the image from the camera, manipulating the image, and performing the VJ method for face detection and eye detection. The important parameters used in the implementation of the VJ method are:

- Haar classifier: the xml file of the Haar classifier for object detection. The classifiers [30]: "haarcascade_frontalface_alt.xml" and "haarcascade_eye_tree_eyeglasses.xml" are employed for the face detection and the eye detection respectively;
- Scale factor: the scale of image resizing each image scale. The default of 1.1 is used in the experiments;
- Minimum size: the minimum size of object allowed by the algorithm. In the experiments, the minimum sizes of 50 $\times$ 50 pixels and 20 $\times$ 20 pixels are used for the face detection and the eye detection respectively.

The eye-map method is implemented using the functions in the OpenCV such as *dilate* for the dilation operation, and *erode* for the erosion operation. Readers may refer to [8,14,19,20] for detailed explanation of the eye-map method. The Hough circle transform method is implemented using the built-in function in OpenCV called as *HoughCircles*.

The serial communication protocol between the face detection module and the eye detection module is implemented in a simple fashion. The data flows in one direction only, i.e., from the face detection module to the eye detection module. The data consists of 13 bytes in ASCII format. The first byte is the character "S" for indicating the start of data packet. The 2^{nd}, 3^{rd}, and 4^{th} bytes are three digits of the x-coordinate of face's bounding box; the 5^{th}, 6^{th}, 7^{th} bytes are three digits of the y-coordinate of face's bounding box; the 8^{th}, 9^{th}, 10^{th} bytes are three digits of the width of face's bounding box; the 11^{th}, 12^{th}, 13^{th} bytes are the three digits of the height of face's bounding box.

3. **Experimental Results.** To validate our methods, two schemes are evaluated. The first scheme is the common or existing method, where the face detection and the eye detection are carried out on one camera system. At first, the face detection is employed to find the face's bounding box. Then the eye detection is applied on the bounding box. Three eye detection methods are evaluated, i.e., using the VJ method [1-4,11,12,16,17] (called as **OCamVJ**), using the eye-map method [8,14,19,20] (called as **OCamMap**), and using the Hough circle transform method [9] (called as **OCamHCT**).

The second scheme is the proposed dual camera system. Six eye detection methods are implemented on the eye detection module, namely as **DCamVJ1** and **DCamVJ2** (using the VJ method), **DCamMap1** and **DCamMap2** (using the eye-map method), **DCamHCT1** and **DCamHCT2** (using the Hough circle transform method). The methods of **DCamVJ1**, **DCamMap1**, and **DCamHCT1** use the resizing images as expressed in (1)-(4). The methods of **DCamVJ2**, **DCamMap2**, and **DCamHCT2** use the resizing images as expressed in (5)-(8), where the fixed height (*sa_height*) in (5) is set to 80 pixels.

To provide a fair comparison, the object (and the movement) captured by the camera should be the same for all scenarios. Unfortunately, it is difficult to ask the people for acting with the same pose and movement repeatedly. Therefore, we record them in the video and play back the video on the computer monitor. The camera system under test is placed in front of the computer monitor. Using this arrangement, it is ensured that all approaches capture the same images during testing.

FIGURE 5. Sample images of the tested videos

There are seven video images used in the experiments as depicted in Figure 5. The images in Figure 5, from left to right, illustrate the sample image frames of tested Video-1 to Video-7 respectively. The faces do not move (or only move slightly) in Video-1, Video-4, Video-5, Video-6, and Video-7. While in Video-2 and Video-3, the faces move from left to right and front to back.

During experiments, three parameters are evaluated, i.e., a) true positive rate (TPR), b) precision (PREC), and c) frame rate (FR). TPR is defined by (9), while PREC is defined by (10). FR is defined as the total numbers of processed image frames of the video in one second, and expressed in fps (frame per second).

$$TPR = TP/(TP + FN) \tag{9}$$

$$PREC = TP/(TP + FP) \tag{10}$$

where TP is the true positive, FN is the false negative, and FP is the false positive.

3.1. **One camera system.** The detection results of one camera system are listed in Table 1. The results show that **OCamVJ** and **OCamHCT** achieve the high TPR, while TPR of **OCamMap** is low. However, since the system (one Raspberry Pi) should compute both the face detection and the eye detection, the frame rate is very low. The fastest FR is achieved by **OCamHCT**, i.e., 3.144 fps. Comparing the precision (PREC) of **OCamVJ** and **OCamHCT**, it is obtained that PREC of **OCamHCT** is lower than **OCamVJ.** It is caused by the fact that **OCamHCT** produces the high false positive (FP), thus according to (10) it yields the low PREC.

TABLE 1. Detection results of one camera system

Video No.	OCamVJ			OCamMap			OCamHCT		
	TPR	**PREC**	**FR (fps)**	**TPR**	**PREC**	**FR (fps)**	**TPR**	**PREC**	**FR (fps)**
Video-1	0.8	1	2.307	0	0	3.056	0.529	0.3913	3.142
Video-2	0.903	1	2.347	0.494	0.5942	3.127	0.931	0.8804	3.119
Video-3	1	0.9767	1.959	0.968	0.581	2.967	1	0.7212	3.089
Video-4	0.842	0.9412	2.056	1	0.9412	3.398	0.974	0.8043	3.420
Video-5	0.75	1	2.282	0.192	1	3.030	1	0.7547	3.115
Video-6	0.941	1	2.085	0.52	0.4643	2.952	1	0.9143	3.056
Video-7	1	0.9524	1.896	0.871	0.4154	2.826	0.902	0.6607	3.062
Average	**0.891**	**0.981**	**2.133**	**0.578**	**0.571**	**3.051**	**0.905**	**0.732**	**3.144**

The results also show that the eye-map method (**OCamMap**) fails to detect eyes in tested Video-1. It achieves the high TPR in several tested videos only (Video-3, Video-4, and Video-7). It could be understood from the characteristic of color segmentation method, which is affected by the color of images.

Typical detection results are depicted in Figure 6, where images in the first, second, and third columns represent the results of **OCamVJ**, **OCamMap**, and **OCamHCT** respectively. The image in the first row of the second column shows that the eye is not detected by **OCamMap**. The false positives in **OCamHCT** are caused by the detected circles of eyebrows as shown in the first and third rows of the third column.

3.2. **Dual camera system.** The detection results of the proposed dual camera system are given in Tables 2 and 3, where the results of **DCamVJ1**, **DCamMap1**, and **DCamHCT1** are listed in Table 2, while the results of **DCamVJ2**, **DCamMap2**, and **DCamHCT2** are listed in Table 3.

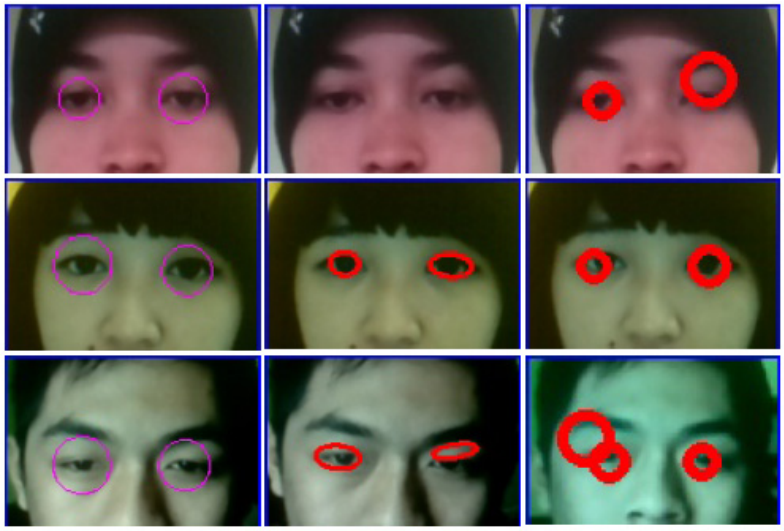

FIGURE 6. Typical images of detection results in the one camera system

TABLE 2. Detection results of **DCamVJ1, DCamMap1, DCamHCT1**

Video No.	DCamVJ1			DCamMap1			DCamHCT1		
	TPR	PREC	FR (fps)	TPR	PREC	FR (fps)	TPR	PREC	FR (fps)
Video-1	0.909	1	6.036	0.767	0.9185	29.369	0.892	0.7363	30.100
Video-2	0.829	1	8.345	0.303	0.6267	30.007	0.494	0.7047	29.813
Video-3	0.832	0.9691	5.413	0.26	0.4726	28.525	0.833	0.5078	30.000
Video-4	1	1	5.045	0.376	0.9412	29.069	0.997	0.7669	30.094
Video-5	0.772	1	6.148	0.93	0.4274	29.985	0.883	0.6811	30.051
Video-6	0.953	1	5.185	0.809	0.4875	30.028	0.994	0.8958	30.132
Video-7	1	1	4.478	0.967	0.4582	29.420	0.981	0.8096	29.576
Average	**0.899**	**0.996**	**5.807**	**0.630**	**0.619**	**29.486**	**0.868**	**0.729**	**29.967**

TABLE 3. Detection results of **DCamVJ2, DCamMap2, DCamHCT2**

Video No.	DCamVJ2			DCamMap2			DCamHCT2		
	TPR	PREC	FR (fps)	TPR	PREC	FR (fps)	TPR	PREC	FR (fps)
Video-1	0.796	1	6.542	0.718	0.7592	30.108	0.938	0.7657	30.098
Video-2	0.899	1	6.942	0.282	0.6183	29.837	0.471	0.7587	29.772
Video-3	0.772	1	7.623	0.29	0.5873	29.769	0.797	0.8366	30.056
Video-4	0.952	0.9677	6.496	0.083	0.2056	30.113	0.912	0.7279	29.983
Video-5	0.881	0.9833	7.537	0.896	0.4117	30.007	0.978	0.7495	30.059
Video-6	0.961	1	6.509	0.588	0.3952	30.132	0.991	0.9913	30.087
Video-7	0.973	1	6.619	0.616	0.3962	30.065	0.995	0.941	30.087
Average	**0.891**	**0.993**	**6.895**	**0.496**	**0.482**	**30.004**	**0.869**	**0.824**	**30.020**

It is interesting to observe the tables that the frame rates of **DCamHCT1** and **DCamHCT2** are very high (about 30 fps), while the true positive rates are also high (about 0.86). The high frame rates are also achieved by **DCamMap1** and **DCamMap2**; however, the true positive rates are low. Comparing the results from the single camera system, it suggests that when the eye detection method is processed separately, the fastest frame rate of 30.020 fps could be achieved. Fortunately, the high true positive rate of 0.869,

and the precision of 0.824 are achieved (see Table 3, **DCamHCT2**). It is worth noting that the Hough circle transform method employed in the proposed dual camera system could detect the eyes very fast with the high true positive rate. The proposed approach also speeds up the computational time of eye detection using the VJ method about three times.

To examine the effectiveness of the proposed dual camera system, we consider two factors in the detection, i.e., the eye detection methods and the resizing methods. The first factor deals with the performances of the VJ, the eye-map, and the Hough transform methods, while the second factor deals with the effectiveness of the resizing method expressed in (1)-(4) (the results are listed in Table 2), and (5)-(8) (the results are listed in Table 3).

The comparisons of TPR, PREC, and FR for all methods are shown in Figure 7. Using the VJ method, TPR and PREC for both the single camera and the dual camera (proposed system) are almost the same. However, FR in the proposed system is three times faster. The significant improvement in FR is achieved by the eye-map and the Hough transform methods, where FR of the proposed system is ten times faster than the existing single camera system, while TPR and PREC only differ slightly. These small differences in TPR are caused by the increase in the numbers of image frames due to the higher frame rate. In some conditions, some images contribute to the false negatives that decrease TPR. In the other conditions, some images contribute to the true positives that increase TPR.

From the results, it could be highlighted that by processing the eye detection on the eye detection module separately, the computational speed could be increased without degrading the true positive rate and the precision significantly. Further the best improvement in the frame rate and the true positive rate is achieved by the Hough circle transform method. The VJ method provides the very high true positive rate, but only small improvement in the frame rate is achieved.

From the experimental results as shown in Figure 7, the image resizing affects the frame rates of the eye-map and the Hough circle transform methods slightly. It increases the frame rates that are lower than 0.5 fps. However, it increases the frame rate of the VJ method in amount of 1 fps. Since the frame rate of the VJ method is about 5 fps, this increment is a significant value. It could be understood that by resizing the image into the smaller size, the VJ is processed faster due to the fact that in the VJ method, the window should be moved over the whole image. Therefore, the smaller image will be processed faster. Contrarily, the eye-map and the Hough circle transform methods process the pixels directly (do not move the window). Thus, the image size does not affect the

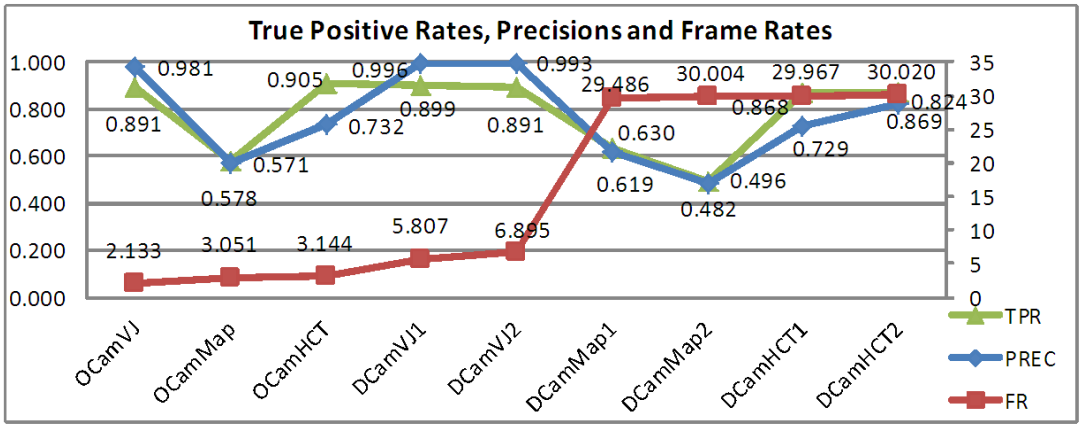

FIGURE 7. Comparisons of true positive rates, precisions, and frame rates

computational time significantly. Since only a few pixels of the image, i.e., the edge pixels or the pixels around the eye are processed, the computational time of the eye-map and the Hough circle transform methods are very fast (about 30 fps).

The resizing image into the smaller image also decreases TPR and PREC of the eye-map method as shown in Figure 7. By observing the experiments, the eye area (the number of pixels) resulting from the eye-map is very small so that it could not be considered as the eye by the algorithm. This factor also affects the VJ and the Hough circle transform methods; however, both methods still work well with the resizing image used in the experiments (i.e., resizing to a fixed height of 80 pixels).

4. **Conclusion.** The implementation of eye detection using dual camera system has been presented in the paper. The proposed system aims to increase the computational speed of the eye detection method by processing the face detection and the eye detection in parallel fashion on the dual camera system. By processing them separately, the eye detection could be processed faster and does not depend on the computational time of the face detection module. This approach works well when the face does not move rapidly. The effectiveness of the proposed system has been validated in the experiments, where the frame rate increases almost ten times compared to the existing methods (single camera system), while the true positive rates and the precisions do not degrade significantly, when the Hough circle transform method is employed in the eye detection module. Further by employing the VJ method in the eye detection module, the true positive rate is very high (almost equal one), but the frame rate is only three times faster than the existing system. The results suggest that the proposed system offers a promising method to be implemented in the real-time application for detecting eye in the driver fatigue detection system. In future, the system will be extended to implement the tracking systems and the other object detection techniques.

Acknowledgment. This work is supported by the Research Grant, Competence-Based Research scheme from Directorate General of Higher Education, Ministry of Research and Technology and Higher Education, Republic of Indonesia (SP/DIPA. 023.04.1.673.453/ 2016). Authors thank people in the Laboratory of Computer Programming and Multimedia, ITN Malang for providing the video data used in the research.

REFERENCES

[1] K. Horak, Fatigue features based on eye tracking for driver inattention system, *Proc. of the 34th Int. Conf. on Telecommunications and Signal Processing*, Budapest, Hungary, pp.593-597, 2011.

[2] T. P. Nguyen, M. T. Chew and S. Demidenko, Eye tracking system to detect driver drowsiness, *Proc. of the 6th Int. Conf. on Automation, Robotics and Application*, Queenstown, New Zealand, pp.472-477, 2015.

[3] Z. B. Ma, Y. Yang, F. Zhou and Z. J. Hua, A real-time fatigue driving detection system design and implementation, *Proc. of the 17th IEEE Int. Conf. on Advanced Communications Technology*, Seoul, South Korea, pp.483-488, 2015.

[4] G. L. R. Clavijo, J. O. Patino and D. M. Leon, Detection of visual fatigue by analyzing the blink rate, *Proc. of the 20th Symp. on Signal Processing, Image and Computer Vision*, Bogota, Colombia, pp.1-5, 2015.

[5] S. Ghosh, T. Nandy and N. Manna, Real time eye detection and tracking method for driver assistance system, in *Lectures Notes in Bioengineering, Advancements of Medical Electronics*, S. Gupta et al. (eds.), Springer India, New Delhi, 2015.

[6] T. H. Chang and Y. R. Chen, Driver fatigue surveillance via eye detection, *Proc. of IEEE the 17th Int. Conf. on Intelligent Transportation Systems*, Qingdao, China, pp.366-371, 2014.

[7] M. J. Flores, J. M. Armingol and A. de la Escalera, Driver drowsiness detection system under infrared illumination for an intelligent vehicle, *IET Intelligent Transportation System*, vol.5, no.4, pp.241-251, 2011.

[8] J. L. C. Bolosan et al., Eye state analysis using EyeMap for drowsiness detection, *Proc. of TENCON-IEEE Region 10 Conference*, Macao, China, pp.1-5, 2015.
[9] B. Alshaqaqi, A. S. Baquhaizel, M. E. A. Quis, M. Boumehed, A. Quamri and M. Keche, Vision based system for driver drowsiness detection, *Proc. of the 11th Int. Symp. Programming and Systems*, Algiers, Algeria, pp.103-108, 2013.
[10] W. C. Cheng, H. C. Liao, M. H. Pan and C. C. Chen, A fatigue detection system with eyeglasses removal, *Proc. of the 15th Int. Conf. on Advanced Communications Technology*, PyeongChang, South Korea, pp.331-335, 2013.
[11] J. F. Xie, M. Xie and W. Zhu, Driver fatigue detection based on head gesture and PERCLOS, *Proc. of Int. Conf. on Wavelet Media Technology and Information Processing*, Chengdu, China, pp.128-131, 2012.
[12] X. Q. Luo, R. Hu and T. E. Fan, The driver fatigue monitoring system based on face recognition technology, *Proc. of the 4th Int. Conf. on Intelligent Control and Information Processing*, Beijing, China, pp.384-388, 2013.
[13] M. Omidyeganeh, A. Javadtalab and S. Shirmohammadi, Intelligent driver drowsiness detection through fusion of yawning and eye closure, *Proc. of IEEE Int. Conf. on Virtual Environments, Human-Computer Interfaces and Measurement Systems*, Ottawa, Canada, pp.1-6, 2011.
[14] R. Ahmed, K. E. K. Emon and M. F. Hossain, Robust driver fatigue recognition using image processing, *Proc. of the 3rd Int. Conf. on Informatics, Electronics and Vision*, Dhaka, Bangladesh, pp.1-6, 2014.
[15] L. Li, Y. Chen and Z. Li, Yawning detection for monitoring driver fatigue based on two cameras, *Proc. of the 12th Int. IEEE Conf. on Intelligent Transportation Systems*, St. Louis Missouri, USA, pp.1-6, 2009.
[16] W. Xu and E. J. Lee, Eye detection and tracking using rectangle features and integrated eye tracker by web camera, *Int. J. of Multimedia and Ubiquitous Engineering*, vol.8, no.4, pp.25-34, 2013.
[17] S. Y. Gwon, C. W. Cho, H. C. Lee, W. O. Lee and K. R. Park, Robust eye and pupil detection method for gaze tracking, *Int. J. of Adv. Robotic Systems*, vol.10, no.98, pp.1-7, 2013.
[18] R. C. Coetzer and G. P. Hancke, Development of a robust active infrared-based eye tracker, *IET Computer Vision*, vol.8, no.6, pp.523-534, 2014.
[19] R. L. Hsu, M. Abdel-Mottaleb and A. K. Jain, Face detection in color images, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.24, no.5, pp.696-706, 2002.
[20] H. Kalbkhani, M. G. Shayesteh and S. M. Mousavi, Efficient algorithms for detection of face, eye and eye state, *IET Computer Vision*, vol.7, no.3, pp.184-200, 2013.
[21] A. Soetedjo, Eye detection based-on color and shape features, *Int. J. of Adv. Computer Sci. and Appl.*, vol.8, no.5, pp.17-22, 2011.
[22] Z. Mei, J. Liu, Z. Li and L. Yang, Study of the eye-tracking methods based on video, *Proc. of the 3rd Int. Conf. on Computational Intelligence, Communication Systems and Networks*, Bali, Indonesia, pp.1-5, 2011.
[23] E. Bilgin and S. Robila, Road sign recognition system on Raspberry Pi, *Proc. of IEEE Long Island Systems, Applications and Technology Conference*, New York, USA, pp.1-5, 2016.
[24] G. Cocorullo, P. Corsonello, F. Frustaci, L. Guachi and S. Perri, Embedded surveillance system using background subtraction and Raspberry Pi, *Proc. of AEIT International Annual Conference*, Naples, Italy, pp.1-5, 2015.
[25] G. G. J. Ramos, J. C. S. Garcia and V. Ponomariov, Embedded system for real-time person detecting in infrared images/videos using super-resolution and Haar-like feature techniques, *Proc. of the 12th International Conference on Electrical Engineering, Computing Science and Automatic Control*, Mexico City, Mexico, pp.1-6, 2015.
[26] P. Viola and M. J. Jones, Robust real-time face detection, *Int. J. of Computer Vision*, vol.57, no.2, pp.137-154, 2004.
[27] *https://www.raspberrypi.org/*.
[28] *https://www.raspbian.org/*.
[29] *http://opencv.org/*.
[30] *https://github.com/opencv/opencv/tree/master/data/haarcascades*.